# Polynomial Synthesis of Asynchronous Automata

Nicolas BAUDRU & Rémi MORIN

Laboratoire d'Informatique Fondamentale de Marseille
39 rue Frédéric Joliot-Curie, F-13453 Marseille cedex 13, France

**Abstract.** Zielonka's theorem shows that each regular set of Mazurkiewicz traces can be implemented as a system of synchronized processes with a distributed control structure called asynchronous automaton. This paper gives a polynomial algorithm for the synthesis of a non-deterministic asynchronous automaton from a regular Mazurkiewicz trace language. This new construction is based on an unfolding approach that improves the complexity of Zielonka's and Pighizzini's techniques in terms of the number of states.

**Keywords**: Concurrency theory, automata, formal languages.



## Introduction

One of the major contributions in the theory of Mazurkiewicz traces [3] characterizes regular languages by means of asynchronous automata [15] which are devices with a distributed control structure. So far all known constructions of asynchronous automata from regular trace languages are quite involved and yield an exponential explosion of the number of states [9]. Furthermore conversions of non-deterministic asynchronous automata into deterministic ones rely on Zielonka's time-stamping function [6, 10] and suffer from the same state-explosion problem. Interestingly heuristics to build small deterministic asynchronous automata were proposed recently in [13].

Zielonka's theorem and related techniques are fundamental tools in concurrency theory. For instance they are useful to compare the expressive power of classical models of concurrency such as Petri nets, asynchronous systems, and concurrent automata [14, 7]. These methods have been adapted already to the construction of communicating finite-state machines from regular sets of message sequence charts [8]. More recently the construction of asynchronous cellular automata [2] was used to implement globally-cooperative high-level message sequence charts [5]. All these constructions yield an exponential explosion of the number of local states.

In this paper we give a *polynomial* construction of non-deterministic asynchronous automata. Our algorithm starts from the specification of a regular trace language in the form of a possibly non-deterministic automaton. The latter is unfolded inductively on the alphabet into an automaton that enjoys several structural properties (Section 2). Next the unfolding automaton is used as the common skeleton of all local processes (Subsection 3.2). Our algorithm is designed specifically to ensure that the number of local states built is polynomial in the number of global states in the specification (Subsection 3.1). We show how this approach subsumes the complexity of Zielonka's and Pighizzini's constructions (Subsection 1.3).

# 1 Background and main result

In this paper we fix a finite alphabet $\Sigma$ provided with a total order $\sqsubseteq$. An automaton over a subset $T \subseteq \Sigma$ is a structure $\mathcal{A} = (Q, \imath, T, \longrightarrow, F)$ where $Q$ is a *finite* set of states, $\imath \in Q$ is an initial state, $\longrightarrow \subseteq Q \times T \times Q$ is a set of transitions, and $F \subseteq Q$ is a subset of final states. We write $q \xrightarrow{a} q'$ to denote $(q, a, q') \in \longrightarrow$. Then the automaton $\mathcal{A}$ is called *deterministic* if we have $q \xrightarrow{a} q' \wedge q \xrightarrow{a} q'' \Rightarrow q' = q''$. For any word $u = a_1 ... a_n \in \Sigma^\star$, we write $q \xrightarrow{u} q'$ if there are some states $q_0, q_1, ..., q_n \in Q$ such that $q = q_0 \xrightarrow{a_1} q_1 ... q_{n-1} \xrightarrow{a_n} q_n = q'$. A state $q \in Q$ is *reachable* if $\imath \xrightarrow{u} q$ for some $u \in \Sigma^\star$. The language $L(\mathcal{A})$ accepted by some automaton $\mathcal{A}$ consists of all words $u \in \Sigma^\star$ such that $\imath \xrightarrow{u} q$ for some $q \in F$. A subset of words $L \subseteq \Sigma^\star$ is *regular* if it is accepted by some automaton.

## 1.1 Mazurkiewicz traces

We fix an *independence relation* $\|$ over $\Sigma$, that is, a binary relation $\| \subseteq \Sigma \times \Sigma$ which is irreflexive and symmetric. For any subset of actions $T \subseteq \Sigma$, the *dependence graph* of $T$ is the undirected graph $(V, E)$ whose set of vertices is $V = T$ and whose edges denote dependence, i.e. $\{a, b\} \in E \Leftrightarrow a \not\| b$.

The *trace equivalence* $\sim$ associated with the independence alphabet $(\Sigma, \|)$ is the least congruence over $\Sigma^\star$ such that $ab \sim ba$ for all pairs of independent actions $a \| b$. For a word $u \in \Sigma^\star$, the *trace* $[u] = \{v \in \Sigma^\star \mid v \sim u\}$ collects all words that are equivalent to $u$. We extend this notation from words to sets of words in a natural way: For all $L \subseteq \Sigma^\star$, we put $[L] = \{v \in \Sigma^\star \mid \exists u \in L, v \sim u\}$.

A *trace language* is a subset of words $L \subseteq \Sigma^\star$ that is closed for trace equivalence: $u \in L \wedge v \sim u \Rightarrow v \in L$. Equivalently we require that $L = [L]$. With no surprise a trace language $L$ is called *regular* if it is accepted by some automaton.

## 1.2 Asynchronous systems vs. asynchronous automata

Two classical automata-based models are known to correspond to regular trace languages. Let us first recall the basic notion of asynchronous systems [1].

DEFINITION 1.1. *An automaton $\mathcal{A} = (Q, \imath, \Sigma, \longrightarrow, F)$ over the alphabet $\Sigma$ is called an* asynchronous system *over $(\Sigma, \|)$ if we have*

ID: $q_1 \xrightarrow{a} q_2 \wedge q_2 \xrightarrow{b} q_3 \wedge a \| b$ *implies* $q_1 \xrightarrow{b} q_4 \wedge q_4 \xrightarrow{a} q_3$ *for some* $q_4 \in Q$.

The Independent Diamond property ID ensures that the language $L(\mathcal{A})$ of any asynchronous system is closed for the commutation of independent adjacent actions. Thus it is a regular trace language. Conversely it is easy to observe that *any regular trace language is the language of some deterministic asynchronous system.*

We recall now a more involved model of communicating processes known as asynchronous automata [15]. A finite family $\delta = (\Sigma_k)_{k \in K}$ of subsets of $\Sigma$ is called a *distribution of* $(\Sigma, \|)$ if we have $a \not\| b \Leftrightarrow \exists k \in K, \{a, b\} \subseteq \Sigma_k$ for all actions $a, b \in \Sigma$. Note that each subset $\Sigma_k$ is a clique of the dependence graph $(\Sigma, \not\|)$ and a distribution $\delta$ is simply a clique covering of $(\Sigma, \not\|)$. We fix an arbitrary distribution $\delta = (\Sigma_k)_{k \in K}$ in the

rest of this paper. We call *processes* the elements of $K$. The *location* $\mathrm{Loc}(a)$ of an action $a \in \Sigma$ consists of all processes $k \in K$ such that $a \in \Sigma_k$: $\mathrm{Loc}(a) = \{k \in K \mid a \in \Sigma_k\}$.

DEFINITION 1.2. *An* asynchronous automaton *over the distribution $(\Sigma_k)_{k\in K}$ consists of a family of finite sets of states $(Q_k)_{k\in K}$, a family of initial local states $(\imath_k)_{k\in K}$ with $\imath_k \in Q_k$, a subset of final global states $F \subseteq \prod_{k\in K} Q_k$, and a transition relation $\partial_a \subseteq \prod_{k\in \mathrm{Loc}(a)} Q_k \times \prod_{k\in \mathrm{Loc}(a)} Q_k$ for each action $a \in \Sigma$.*

The set of *global states* $Q = \prod_{k\in K} Q_k$ can be provided with a set of global transitions $\longrightarrow$ in such a way that an asynchronous automaton is viewed as a particular automaton. Given an action $a \in \Sigma$ and two global states $q = (q_k)_{k\in K}$ and $r = (r_k)_{k\in K}$, we put $q \xrightarrow{a} r$ if $\left((q_k)_{k\in \mathrm{Loc}(a)}, (r_k)_{k\in \mathrm{Loc}(a)}\right) \in \partial_a$ and $q_k = r_k$ for all $k \in K \setminus \mathrm{Loc}(a)$. The initial global state $\imath$ consists of the collection of initial local states: $\imath = (\imath_k)_{k\in K}$. Then the *global automaton* $\mathcal{A} = (Q, \imath, \Sigma, \longrightarrow, F)$ satisfies Property ID of Def. 1.1. Thus it is an asynchronous system over $(\Sigma, \|)$ and $L(\mathcal{A})$ is a regular trace language. An asynchronous automaton is *deterministic* if its global automaton is deterministic, i.e. the local transition relations $\partial_a$ are partial functions.

### 1.3 Main result and comparisons to related works

Although deterministic asynchronous automata appear as a restricted subclass of deterministic asynchronous systems, Zielonka's theorem asserts that any regular trace language can be implemented in the form of a deterministic asynchronous automaton.

THEOREM 1.3. *[15] For any regular trace language $L$ there exists a deterministic asynchronous automaton whose global automaton $\mathcal{A}$ satisfies $L = L(\mathcal{A})$.*

In [9] a complexity analysis of Zielonka's construction is detailed. Let $|Q|$ be the number of states of the minimal deterministic automaton that accepts $L$ and $|K|$ be the number of processes. Then the number of local states built by Zielonka's technique in each process $k \in K$ is $|Q_k| \leqslant 2^{O(2^{|K|}.|Q|\log|Q|)}$. The simplified construction by Cori et al. in [2] also suffers from this exponential state-explosion [3].

Another construction proposed by Pighizzini [12] builds some non-deterministic asynchronous automata from particular rational expressions that refine Ochmański's theorem [11]. This simpler approach proceeds inductively on the structure of the rational expression. Each step can easily be shown to be polynomial. In particular the number of local states in each process is (at least) *doubled* by each restricted iteration. Consequently in some cases the number of local states in each process is *exponential* in the length of the rational expression.

In the present paper we give a new construction that is *polynomial in* $|Q|$ (Th. 3.1): It produces $|Q_k| \leqslant O(|Q|^d)$ local states for each process, where $d = (2.|\Sigma| + 2)^{|\Sigma|+1}$, $|\Sigma|$ is the size of $\Sigma$, and $|Q|$ is the number of states of some (possibly non-deterministic) asynchronous system that accepts $L$. Noteworthy the number of local states $|Q_k|$ obtained by our approach is independent from the number of processes $|K|$.

## 2 Unfolding algorithm

In the rest of the paper we fix some automaton $\mathcal{A} = (Q, \imath, \Sigma, \longrightarrow, F)$ that is possibly non-deterministic. The aim of this section is to associate with $\mathcal{A}$ a family of automata called *boxes* and *triangles* which are defined inductively. The last box built by this construction will be called the *unfolding* of $\mathcal{A}$ (Def. 2.3).

Boxes and triangles are related to $\mathcal{A}$ by means of morphisms which are defined as follows. Let $\mathcal{A}_1 = (Q_1, \imath_1, T, \longrightarrow_1, F_1)$ and $\mathcal{A}_2 = (Q_2, \imath_2, T, \longrightarrow_2, F_2)$ be two automata over a subset of actions $T \subseteq \Sigma$. A *morphism* $\sigma : \mathcal{A}_1 \to \mathcal{A}_2$ *from* $\mathcal{A}_1$ *to* $\mathcal{A}_2$ is a mapping $\sigma : Q_1 \to Q_2$ from $Q_1$ to $Q_2$ such that $\sigma(\imath_1) = \imath_2$, $\sigma(F_1) \subseteq F_2$, and $q_1 \xrightarrow{a}_1 q_1'$ implies $\sigma(q_1) \xrightarrow{a}_2 \sigma(q_1')$. In particular, $L(\mathcal{A}_1) \subseteq L(\mathcal{A}_2)$.

Now boxes and triangles are associated with an initial state that may not correspond to the initial state of $\mathcal{A}$. They are associated also with a subset of actions $T \subseteq \Sigma$. For these reasons, for any state $q \in Q$ and any subset of actions $T \subseteq \Sigma$, we let $\mathcal{A}_{T,q}$ denote the automaton $(Q, q, T, \longrightarrow_T, F)$ where $\longrightarrow_T$ is the restriction of $\longrightarrow$ to the transitions labeled by actions in $T$: $\longrightarrow_T = \longrightarrow \cap (Q \times T \times Q)$.

In this section we shall define the box $\square_{T,q}$ for all states $q \in Q$ and all subsets of actions $T \subseteq \Sigma$. The box $\square_{T,q}$ is a pair $(\mathcal{B}_{T,q}, \beta_{T,q})$ where $\mathcal{B}_{T,q}$ is an automaton over $T$ and $\beta_{T,q} : \mathcal{B}_{T,q} \to \mathcal{A}_{T,q}$ is a morphism. Similarly, we shall define the triangle $\triangle_{T,q}$ for all states $q$ and all *non-empty* subsets of transitions $T$. The triangle $\triangle_{T,q}$ is a pair $(\mathcal{T}_{T,q}, \tau_{T,q})$ where $\mathcal{T}_{T,q}$ is an automaton over $T$ and $\tau_{T,q} : \mathcal{T}_{T,q} \to \mathcal{A}_{T,q}$ is a morphism.

The *height* of a box $\square_{T,q}$ or a triangle $\triangle_{T,q}$ is the cardinality of $T$. Boxes and triangles are defined inductively. We first define the box $\square_{\emptyset,q}$ for all states $q \in Q$. Then triangles of height $h$ are built upon boxes of height $g < h$ and boxes of height $h$ are built upon either triangles of height $h$ or boxes of height $g < h$, whether the dependence graph $(T, \not\parallel)$ is connected or not.

The base case deals with the boxes of height 0. For all states $q \in Q$, the box $\square_{\emptyset,q}$ consists of the morphism $\beta_{\emptyset,q} : \{q\} \to Q$ that maps $q$ to itself together with the automaton $\mathcal{B}_{\emptyset,q} = (\{q\}, q, \emptyset, \emptyset, F_{\emptyset,q})$ where $F_{\emptyset,q} = \{q\}$ if $q \in F$ and $F_{\emptyset,q} = \emptyset$ otherwise. More generally a state of a box or a triangle is final if it is associated with a final state of $\mathcal{A}$.

### 2.1 Building triangles from boxes

Triangles are made of boxes of lower height. Boxes are inserted into a triangle inductively on the height along a tree-like structure and several copies of the same box may appear within a triangle. We want to keep track of this structure in order to prove properties of triangles (and boxes) inductively. This enables us also to allow for the distinction of different copies of the same box within a triangle.

To do this, each state of a triangle is associated with a *rank* $k \in \mathbb{N}$ such that all states with the same rank come from the same copy of the same box. It is also important to keep track of the height each state comes from, because boxes of a triangle are included inductively on the height. For these reasons, a state of a triangle $\triangle_{T^\circ,q^\circ} = (\mathcal{T}_{T^\circ,q^\circ}, \tau_{T^\circ,q^\circ})$ is encoded as a quadruple $v = (w, T, q, k)$ such that $w$ is a state from the box $\square_{T,q}$ with height $h = |T|$ and $v$ is added to the triangle within the $k$-th box inserted into the triangle. Moreover this box is a copy of $\square_{T,q}$. In that case

```
BUILD-TRIANGLE(T°, q°)
 1  (B, β) ← BUILD-BOX(∅, q°)
 2  (T, τ) ← MARK((B, β), ∅, q°, 1)
 3  k ← 1
 4  for h ← 1 to |T°| − 1
 5  do for v = (w, T, q, l)  a state of  T  with  |T| = h − 1
 6     do for q' ∈ Q and a ∈ T° \ T
 7        do if β_{T,q}(w) —a→ q'
 8              then T' ← T ∪ {a}                          Here |T'| = h < |T°|
 9                   (B, β) ← BUILD-BOX(T', q')            Compute □_{T',q'}
10                   k ← k + 1
11                   (B', β') ← MARK((B, β), T', q', k)    Mark it with T', q', k
12                   INSERT((T, τ), (B', β'))              Insert it into (T, τ)
13                   ADD((T, τ), (v, a, (ι_{□,T',q'}, T', q', k)))
14  return (T, τ)
```

N.B. Line 12, $\imath_{\square,T',q'}$ denote the initial state of the box $\square_{T',q'}$.

**Alg. 1.** Construction of a triangle

the state $v$ maps to $\tau_{T^\circ,q^\circ}(v) = \beta_{T,q}(w)$, that is, the insertion of boxes preserves the correspondance to the states of $\mathcal{A}$. Moreover the morphism $\tau_{T^\circ,q^\circ}$ of a triangle $\triangle_{T^\circ,q^\circ}$ is encoded in the data structure of its states.

The construction of the triangle $\triangle_{T^\circ,q^\circ}$ is detailed in Algorithm 1. It relies on four procedures:

- BUILD-BOX$(T, q)$ returns the box $\square_{T,q}$.
- MARK$((\mathcal{B}, \beta), T, q, k)$ returns a copy of $(\mathcal{B}, \beta)$ where each state $w$ from $\mathcal{B}$ is replaced by the marked state $v = (w, T, q, k)$.
- INSERT$((\mathcal{T}, \tau), (\mathcal{B}, \beta))$ inserts $(\mathcal{B}, \beta)$ within $(\mathcal{T}, \tau)$; the initial state of this disjoint union of automata is the initial state of $(\mathcal{T}, \tau)$.
- ADD$((\mathcal{T}, \tau), (v, a, v'))$ adds a new transition $v \xrightarrow{a} v'$ to the automaton $\mathcal{T}$; it is required that $v$ and $v'$ be states of $\mathcal{T}$.

The construction of the triangle $\triangle_{T^\circ,q^\circ}$ starts with building a copy of the base box $\square_{\emptyset,q^\circ}$ which gets rank $k = 1$ and whose marked initial state $(\imath_{\square,\emptyset,q^\circ}, \emptyset, q^\circ, 1)$ becomes the initial state of $\triangle_{T^\circ,q^\circ}$. Along the construction of this triangle, $k$ counts the number of boxes already inserted in the triangle. The insertion of boxes proceeds inductively on the height $h$ (Line 4) as follows: For each state $v = (w, T, q, l)$ with height $|T| = h-1$, if a transition $\beta_{T,q}(w) \xrightarrow{a} q'$ in $\mathcal{A}$ carries an action $a \in T^\circ \setminus T$ (Line 6) then a new box $\square_{T',q'}$ of height $h$ is inserted with $T' = T \cup \{a\}$ (Line 12) and a transition $v \xrightarrow{a} v'$ is added to the triangle $\mathcal{T}_{T^\circ,q^\circ}$ in construction (Line 13) where $v'$ is the marked initial state of the new box $\square_{T',q'}$. We stress here that $\tau(v) \xrightarrow{a} \tau(v')$ is a transition of $\mathcal{A}_{T^\circ,q^\circ}$ because $\tau(v) = \beta_{T,q}(w)$ and $\tau(v') = \beta_{T',q'}(\imath_{\square,T',q'}) = q'$. This observation will show that $\tau$ is a morphism. Another useful remark is the following.

LEMMA 2.1. *If a word $u \in \Sigma^\star$ leads in the triangle $\triangle_{T^\circ,q^\circ}$ from its initial state $(\imath_{\square,\emptyset,q^\circ}, \emptyset, q^\circ, 1)$ to some state $v = (w, T, q, l)$ then each action of $T$ occurs in $u$.*

### 2.2 Building boxes from triangles

We distinguish two cases when we build the box $\square_{T,q}$ whether the dependence graph $(T, \not\parallel)$ is connected or not. In case $(T, \not\parallel)$ is a connected graph then the box $\square_{T,q}$ collects all triangles $\triangle_{T,q'}$ for all states $q' \in Q$. Each triangle is duplicated a fixed number of times and copies of triangles are connected in some particular way. Similarly to triangles, the states of a box are decorated with a rank $k$ that distinguishes states from different triangles and also states from different copies of the same triangle. We adopt the same data structure as for triangles: A state $v$ of a box is a quadruple $(w, T, q, k)$ where $w$ is a state of $\triangle_{T,q}$ and $k \in \mathbb{N}$. Whereas triangles of height $h$ are built upon boxes of height $g < h$, boxes $\square_{T,q}$ are built upon triangles $\triangle_{T,q'}$ with the same set of transitions $T$ — and consequently, with the same height. Similarly to the algorithm BUILD-TRIANGLE, the algorithm that builds boxes uses an integer variable $k$ that counts the number of triangles already inserted in the box in construction.

In case the dependence graph $(T, \not\parallel)$ is not connected, we let $T_1$ denote the connected component of $(T, \not\parallel)$ that contains the least action $a \in T$ w.r.t. the total order $\sqsubseteq$ over $\Sigma$ and we put $T_2 = T \setminus T_1$. Then the box $\square_{T,q}$ is built upon a copy of the box $\square_{T_2,q}$ connected to copies of boxes $\square_{T_1,q_1}$ for some states $q_1 \in Q$.

The construction of the box $\square_{T^\circ,q^\circ}$ is detailed in Algorithm 5. It relies on ten procedures:

- BASE-BOX$(q)$ returns the base box $\square_{\emptyset,q}$.
- EMPTY-BOX returns a special new box called *empty box*.
- MARK, INSERT and ADD are the procedures used for BUILD-TRIANGLE. If $(\mathcal{B}, \beta)$ is this special empty box then INSERT$((\mathcal{B}, \beta), (\mathcal{T}, \tau))$ replaces simply $(\mathcal{B}, \beta)$ by $(\mathcal{T}, \tau)$.
- MISSING$(T^\circ, q, q')$ returns the set of all pairs $(w, a)$ where $w$ is a state that has been inserted in the triangle $\triangle_{T^\circ,q}$ within a box $\square_{T'',q''}$ such that $|T''| = |T^\circ| - 1$ and the action $a \in T^\circ \setminus T''$ is such that there is a transition $\tau_{T^\circ,q}(w) \xrightarrow{a} q'$ in $\mathcal{A}$ (Alg. 2). Due to the structure of triangles, if $(w, a)$ is a missing transition then there is no transition $w \xrightarrow{a}_{\triangle,T^\circ,q} w'$ with $\tau_{T^\circ,q}(w') = q'$ in $\triangle_{T^\circ,q}$.
- MIN-RANK$(T^\circ, q, \mathcal{B}, k)$ returns the minimal rank of a copy of a triangle $\mathcal{T}_{T^\circ,q}$ inserted in $\mathcal{B}$ where $k$ is the maximal rank of triangles in $\mathcal{B}$ (Alg. 3).
- MAX-OUT-DEGREE$(T^\circ)$ returns the number of copies of each triangle $\triangle_{T^\circ,q}$ that should compose the box $\square_{T^\circ,q^\circ}$. It does not depend on $q$ but it depends on the

```
MISSING(T°, q, q')
1  M ← ∅
2  (T, τ) ← BUILD-TRIANGLE(T°, q)
3  for w ∈ Q_{△,T°,q} such that w = (w'', T'', q'', k'') and |T''| = |T°| − 1
4  do if τ_{T°,q}(w) —a→ q' with a ∈ T° \ T''
5        then M ← M ∪ {(w, a)}
6  return (M)
```

N.B. The triangle $\triangle_{T^\circ,q} = (\mathcal{T}, \tau)$ computed at Line 2 consists of a set of states $Q_{\triangle,T^\circ,q}$ and a transition relation $\longrightarrow_{\triangle,T^\circ,q}$.

**Alg. 2.** Set of missing transitions from a triangle $\triangle_{T^\circ,q}$ to some state $q'$

```
MIN-RANK(T°, q, B, k)
1  f ← k + 1
2  for v = (w, T', q', l) in B
3  do if q' = q and T' = T°
4        then if l < f
5                then f = l
6  return (f)
```

**Alg. 3.** Minimal rank of $\triangle_{T^\circ,q}$ in $\mathcal{B}$

```
MAX-OUT-DEGREE(T°)
1  m ← 0
2  for q, q' ∈ Q
3  do n ← |MISSING(T°, q, q')|
4     if n > m
5        then m ← n
6  return (m)
```

**Alg. 4.** Minimal number of copies required

cardinality of all sets $\textsc{Missing}(T^\circ, q, q')$ with $q, q' \in Q$ (Alg. 4). The rôle of these copies is detailed below.

- $\textsc{Clean}(\mathcal{B}, \beta)$ remove all unreachable states from $\mathcal{B}$.
- $\textsc{Decomposition}(T^\circ)$ returns the connected component $T$ of $(T^\circ, \not\parallel)$ that contains the minimal action of $T^\circ$ w.r.t. the total order $\sqsubseteq$.

The construction of the box $\square_{T^\circ,q^\circ}$ starts with solving the base case where $T^\circ = \emptyset$ (Line 1). Assume now that the dependence graph $(T^\circ, \not\parallel)$ is connected (Line 4). Then the box is initialized as the special empty box (Line 6). The number $m$ of copies of each triangle $\triangle_{T^\circ,q}$ is computed in Line 8 with the help of functions MAX-OUT-DEGREE and MISSING. Next these copies are inserted and the first copy of $\triangle_{T^\circ,q^\circ}$ gets rank $k = 1$ (Lines 9 to 14). Consequently the initial state of the box $\square_{T^\circ,q^\circ}$ in construction is the first copy of the initial state $\imath_{\triangle,T^\circ,q^\circ}$ of the triangle $\triangle_{T^\circ,q^\circ}$, that is: $(\imath_{\triangle,T^\circ,q^\circ}, T^\circ, q^\circ, 1)$. Noteworthy copies of the same triangle have consecutive ranks.

In a second step transitions are added to connect these triangles to each other (Lines 15 to 25). Intuitively a $a$-transition is *missing* from the state $w = (w'', T'', q'', k'')$ of the triangle $\triangle_{T^\circ,q}$ to the state $q'$ of $\mathcal{A}$ if $|T^\circ \setminus T''| = 1$ — i.e. this state has been inserted at the highest level in $\triangle_{T^\circ,q}$ — and there exists in $\mathcal{A}$ a transition $\tau_{T^\circ,q}(w) \xrightarrow{a} q'$ with $a \in T^\circ \setminus T''$ but no transition $w \xrightarrow{a} w'$ with $\tau_{T^\circ,q}(w') = q'$ in $\triangle_{T^\circ,q}$.

The rôle of MISSING is to compute the missing transitions w.r.t. $q$, $q'$, and $T^\circ$. For each such missing transition $(w, a)$ we connect each copy of $w$ to the initial state of a copy of $\triangle_{T^\circ,q'}$. In this process we require two crucial properties:

$\mathsf{P}_1$: No added transition connects two states from the same copy of the same triangle: $(w, T^\circ, q, l)$ should not be connected to $(\imath_{\triangle,T^\circ,q}, T^\circ, q, l)$.

$\mathsf{P}_2$: At most one transition connects one copy of $\triangle_{T^\circ,q}$ to one copy of $\triangle_{T^\circ,q'}$: If we add from a given copy of $\triangle_{T^\circ,q}$ a transition $(w_1, T^\circ, q, l) \xrightarrow{a} (\imath_{\triangle,T^\circ,q'}, T^\circ, q', l')$ and a transition $(w_2, T^\circ, q, l) \xrightarrow{b} (\imath_{\triangle,T^\circ,q'}, T^\circ, q', l')$ to the same copy of $\triangle_{T^\circ,q'}$ then $w_1 = w_2$ and $a = b$.

The minimal number of copies required to fulfill these conditions is computed by MAX-OUT-DEGREE. For a fixed missing transition $(w, a)$ from a state $w$ of the triangle $\triangle_{T^\circ,q}$ to a state $q'$ of $\mathcal{A}$, Lines 22 to 25 add a transition from the $j$-th copy of $w$ to the $c$-th copy of the initial state of $\triangle_{T^\circ,q'}$ with the property that $j \neq c$ if $q = q'$ (Condition $\mathsf{P}_1$ above). Moreover states from the $j$-th copy of $\triangle_{T^\circ,q}$ are connected to distinct copies of the initial state of $\triangle_{T^\circ,q'}$ (Condition $\mathsf{P}_2$ above).

Note here that each new transition $(v, a, v')$ added to $(\mathcal{B}, \beta)$ at Line 25 is such that $\beta(v) \xrightarrow{a} \beta(v')$ is a transition from $\mathcal{A}_{T^\circ,q^\circ}$ because $\beta(v) = \tau_{T^\circ,q}(w)$, $\beta(v') =$

$\tau_{T^\circ,q'}(\imath_{\triangle,T^\circ,q'}) = q'$, and $\tau_{T^\circ,q}(w) \xrightarrow{a} q'$. Again, this observation will show that $\beta$ is a morphism. A crucial remark for boxes of connected alphabets is the following.

LEMMA 2.2. *If a non-empty word $u$ leads from the initial state of a triangle $\triangle_{T^\circ,q}$ to the initial state of a triangle $\triangle_{T^\circ,q'}$ within the box $\square_{T^\circ,q^\circ}$ then each action of $T^\circ$ occurs in $u$.*

For simplicity's sake our algorithm uses the same number of copies for each triangle. This approach yields in general unreachable states in useless copies. The latter are removed by CLEAN at Line 26.

Assume now that $(T^\circ, \not\parallel)$ is not connected (Line 29). Let $T_1$ be the connected component of $T^\circ$ that contains the least action of $T^\circ$ w.r.t. the total order $\sqsubseteq$ over $\Sigma$. We put $T_2 = T^\circ \setminus T_1$. The construction of the box $\square_{T^\circ,q^\circ}$ starts with building a copy of the box $\square_{T_2,q^\circ}$. Next for each state $w$ of $\square_{T_2,q^\circ}$ and each transition $\beta_{T_2,q}(w) \xrightarrow{a} q'$ with $a \in T_1$, the algorithm inserts a (new) copy of the box $\square_{T_1,q'}$ and adds a transition from the copy of $w$ to the initial state of the copy of $\square_{T_1,q'}$. By recursive calls of BUILD-BOX the box $\square_{T^\circ,q}$ is built along a tree-like structure upon copies of boxes $\square_{T',q'}$ where $T'$ is a connected component of $T^\circ$.

### 2.3 Remarks

From a mathematical viewpoint, Algorithms 1 to 5 are meant to define boxes $\square_{T,q}$ and triangles $\triangle_{T,q}$. Thus two instances of BUILD-TRIANGLE$(T, q)$ produce the same object. For this reason, we speak of *the* triangle $\triangle_{T,q}$. This is particularly important to understand the interaction between BUILD-BOX and MISSING. In case $T$ is connected, Algorithm BUILD-BOX proceeds in two steps. First several copies of each triangle $\triangle_{T,q}$ are collected and next some transitions are added from some states of copies of $\triangle_{T,q}$ to the initial state of copies of $\triangle_{T,q'}$. These additional transitions are computed in a separate function MISSING that depends on triangles. It is crucial that the triangles $\triangle_{T,q}$ used by the function MISSING be the same as the triangles $\triangle_{T,q}$ inserted in BUILD-BOX.

From a more computational viewpoint, Algorithms 1 to 5 can obviously be implemented. To do this, we require that each triangle and each box be constructed only once. An alternative to this requirement is to adapt the parameters of the function MISSING and ensure that BUILD-BOX transfers its own triangle $\triangle_{T,q}$ instead of the pair $(T, q)$ to that function so that the set of states computed by MISSING matches the set of states used by BUILD-BOX. However it need not to transfert its own triangle $\triangle_{T,q}$ to the function MAX-OUT-DEGREE because this function works on triangles up to isomorphisms.

In this section we have built a family of boxes and triangles from a fixed automaton $\mathcal{A}$. This construction leads us to the definition of the unfolding of $\mathcal{A}$ as follows.

DEFINITION 2.3. *The* unfolding $\mathcal{A}_{\mathrm{Unf}}$ *of the automaton $\mathcal{A} = (Q, \imath, \Sigma, \longrightarrow, F)$ is the box $\mathcal{B}_{\Sigma,\imath}$; moreover $\beta_{\mathrm{Unf}}$ denote the mapping $\beta_{\Sigma,\imath}$ from the states of $\mathcal{A}_{\mathrm{Unf}}$ to $Q$.*

In the next section we study some complexity, structural, and semantical properties of this object. We assume that $\mathcal{A}$ satisfies Property ID of Definition 1.1 so that it accepts a regular trace language $L$. We explain how to build from the unfolding $\mathcal{A}_{\mathrm{Unf}}$ a nondeterministic asynchronous automaton that accepts $L(\mathcal{A})$.

```
BUILD-BOX(T°, q°)
 1  if T° = ∅
 2    then                                          This is the base case
 3         return (BASE-BOX(q°))
 4  if T° is connected (and non-empty)
 5    then
 6         (B, β) ← EMPTY-BOX                       Initialise (B, β) to be
 7         k ← 0                                    the special empty box
 8         m ← MAX-OUT-DEGREE(T°) + 1
 9         for q ∈ Q  starting with  q°
10         do (T, τ) ← BUILD-TRIANGLE(T°, q)        Compute △_{T°,q}
11            for l ← 1 to m
12            do k ← k + 1                          Insert m copies
13               (T', τ') ← MARK((T, τ), T°, q, k)  Marked with T°, q, k
14               INSERT((B, β), (T', τ'))
15         for q, q' ∈ Q
16         do M ← MISSING(T°, q, q')                List of missing transitions
17            f ← MIN-RANK(T°, q, B, k) − 1         Minimal rank of △_{T°,q}
18            f' ← MIN-RANK(T°, q', B, k) − 1       Minimal rank of △_{T°,q'}
19            for j ← 1 to m
20            do c ← 0                              We have |M| + 1 ⩽ m
21               for (w, a) ∈ M
22               do c ← c + 1                       If q = q' then f = f'
23                  if f + j = f' + c
24                    then c ← c + 1                We have c ⩽ m
25                  ADD((B, β), ((w, T°, q, f + j), a, (ι_{△,T°,q'}, T°, q', f' + c)))
26         CLEAN(B, β)
27         return (B, β)
28  if T° is not connected (nor empty)
29    then
30         T1 ← DECOMPOSITION(T°)
31         T2 ← T° \ T1
32         (B0, β0) ← BUILD-BOX(T2, q°)
33         (B, β) ← MARK((B0, β0), T2, q°, 1)
34         k ← 1
35         for w ∈ Q_{□,T2,q°} , q' ∈ Q  and  a ∈ T1
36         do if β0(w) --a--> q'
37              then k ← k + 1                       Insert a copy of □_{T1,q'}
38                   (B', β') ← BUILD-BOX(T1, q')
39                   (B'', β'') ← MARK((B', β'), T1, q', k)
40                   INSERT((B, β), (B'', β''))
41                   ADD((B, β), ((w, T2, q°, 1), a, (ι_{□,T1,q'}, T1, q', k)))
42         return (B, β)
```

N.B.

- In Line 25 $\imath_{\triangle,T^\circ,q'}$ denotes the initial state of $\triangle_{T^\circ,q'}$.
- In Line 35 $Q_{\square,T_2,q^\circ}$ denotes the set of states of $\square_{T_2,q^\circ}$.
- In Line 41 $\imath_{\square,T_1,q'}$ denotes the initial state of $\square_{T_1,q'}$.

**Alg. 5.** Construction of a box

## 3 Properties of the unfolding algorithm

In this section we fix a regular trace language $L$ over the independence alphabet $(\Sigma, \|)$. We assume that the possibly non-deterministic automaton $\mathcal{A}$ fulfills Property ID of Def. 1.1 and satisfies $L(\mathcal{A}) = L$. First we sketch a complexity analysis of the number of states in the unfolding $\mathcal{A}_{\mathrm{Unf}}$. Next we show in Subsection 3.2 how to build from $\mathcal{A}_{\mathrm{Unf}}$ an asynchronous automaton whose global automaton accepts $L(\mathcal{A})$.

### 3.1 Complexity analysis

For all naturals $n \geqslant 0$ we denote by $\beta_n$ the maximal number of states in a box $\mathcal{B}_{T,q}$ with $|T| = n$ and $q \in Q$. Similarly for all naturals $n \geqslant 1$ we denote by $\tau_n$ the maximal number of states in a triangle $\mathcal{T}_{T,q}$ with $|T| = n$ and $q \in Q$. Noteworthy $\beta_0 = 1$ and $\tau_1 = 1$. Moreover $\tau_n$ is non-decreasing because the triangle $\triangle_{T',q}$ is a subautomaton of the triangle $\triangle_{T,q}$ as soon as $T' \subseteq T$. In the following we assume $2 \leqslant n \leqslant |\Sigma|$.

Consider some subset $T \subseteq \Sigma$ with $|T| = n$. Each triangle $\mathcal{T}_{T,q}$ is built inductively upon boxes of height $h \leqslant n-1$ (see Alg. 1). We distinguish two cases. First boxes of height $h < n-1$ are inserted. Each of these boxes appears also in some triangle $\mathcal{T}_{T',q}$ with $T' \subset T$ and $|T'| = n-1$. Each of these triangles is a subautomaton of $\mathcal{T}_{T,q}$ with at most $\tau_{n-1}$ states. Moreover there are only $n$ such triangles which give rise to at most $n.\tau_{n-1}$ states built along this first step. Second, boxes of height $n-1$ are inserted and connected to states inserted at height $n-2$. Each of these states belongs to some box $\square_{T',q'}$ with $|T'| = n-2$; it gives rise to at most $2.|Q|$ boxes at height $n-1$ because $|T \setminus T'| = 2$: This yields at most $2.|Q|.\beta_{n-1}$ new states. Altogether we get

$$\tau_n \leqslant n.\tau_{n-1}.(1 + 2.|Q|.\beta_{n-1}) \leqslant |\Sigma|.\tau_{n-1}.3.|Q|.\beta_{n-1} \tag{1}$$

Consider now a connected subset $T \subseteq \Sigma$ with $|T| = n-1$. Then each box $\mathcal{B}_{T,q}$ is built upon triangles $\mathcal{T}_{T,q'}$ of height $n-1$ (see Alg. 5). We can check that the value $m =$MAX-OUT-DEGREE$(T)$ is at most $\tau_{n-1} + 1$. Therefore the box $\mathcal{B}_{T,q}$ contains at most $2.\tau_{n-1}$ copies of each triangle $\mathcal{T}_{T,q'}$. Hence $\beta_{n-1} \leqslant 2.|Q|.\tau_{n-1}^2$.

Consider now a non-connected subset $T \subseteq \Sigma$ with $|T| = n-1$. Then each box $\mathcal{B}_{T,q}$ is built upon copies of boxes $\mathcal{B}_{T',q'}$ where $T'$ is a connected component of $(T, \not\|)$. These boxes are inserted inductively along recursive calls of BUILD-BOX and they are connected in a tree-like structure. Each of these boxes contains at most $2.|Q|.\tau_{n-2}^2$ states as explained above. From each state of these boxes at most $(n-2).|Q|$ new boxes are connected. Thus each box $\mathcal{B}_{T',q'}$ is connected to at most $c = |\Sigma|.2.|Q|^2.\tau_{n-2}^2$ boxes in the tree-like structure. Consequently there are at most $1 + c + c^2 + c^3 + ... + c^{n-2}$ boxes. It follows that

$$\beta_{n-1} \leqslant c^{n-1}.2.|Q|.\tau_{n-2}^2 \leqslant 2^n.|\Sigma|^{n-1}.|Q|^{2.n-1}.\tau_{n-2}^{2.n} \tag{2}$$

Since $\tau_{n-2} \leqslant \tau_{n-1}$ we get in both cases $\beta_{n-1} \leqslant 2^{|\Sigma|}.|\Sigma|^{|\Sigma|-1}.|Q|^{2.|\Sigma|-1}.(\tau_{n-1})^{2.|\Sigma|}$.

We can now apply (1) and get $\tau_n \leqslant N.\tau_{n-1}^d$ where $N = 3.2^{|\Sigma|}.|\Sigma|^{|\Sigma|}.|Q|^{2.|\Sigma|}$ and $d = 2.|\Sigma| + 1$. Since $\tau_1 = 1$, we get $\tau_n \leqslant N^{d^{n-1}}$. We can apply (2) with $n$ instead of $n-1$ and get $\beta_n \leqslant 2.|Q|.N.(\tau_n)^{2.(n+1)} \leqslant 2.|Q|.N.N^{d^{n-1}.(2n+2)}$. Finally we have

$$\beta_{|\Sigma|} \leqslant 2.|Q|.\left(3.2^{|\Sigma|}.|\Sigma|^{|\Sigma|}.|Q|^{2.|\Sigma|}\right)^{(2.|\Sigma|+2)^{|\Sigma|}} \in O\left(|Q|^{(2.|\Sigma|+2)^{|\Sigma|+1}}\right) \tag{3}$$

### 3.2 Construction of an asynchronous automaton

Finally we build from the unfolding $\mathcal{A}_{\mathrm{Unf}} = (Q_{\mathrm{Unf}}, \imath_{\mathrm{Unf}}, \Sigma, \longrightarrow_{\mathrm{Unf}}, F_{\mathrm{Unf}})$ of $\mathcal{A}$ an asynchronous automaton $\widehat{\mathcal{A}_{\mathrm{Unf}}}$ that accepts $L(\mathcal{A})$. We define $\widehat{\mathcal{A}_{\mathrm{Unf}}}$ as follows. First we put $Q_k = Q_{\mathrm{Unf}}$ for each process $k \in K$. Next the initial state is the $|K|$-tuple $(\imath_{\mathrm{Unf}}, ..., \imath_{\mathrm{Unf}})$. Moreover for each action $a$, the pair $((q_k)_{k\in \mathrm{Loc}(a)}, (r_k)_{k\in \mathrm{Loc}(a)})$ belongs to the transition relation $\partial_a$ if there exist two states $q, r \in Q_{\mathrm{Unf}}$ and a transition $q \xrightarrow{a}_{\mathrm{Unf}} r$ in the unfolding such that the two following conditions are satisfied:

- for all $k \in \mathrm{Loc}(a)$, $q_k \xrightarrow{u}_{\mathrm{Unf}} q$ for some word $u \in (\Sigma \setminus \Sigma_k)^\star$;
- for all $k \in \mathrm{Loc}(a)$, $r_k = r$; in particular all $r_k$ are equal.

Finally, a global state $(q_k)_{k\in K}$ is *final* if there exists a final state $q \in Q_{\mathrm{Unf}}$ such that for all $k \in K$ there exists a path $q_k \xrightarrow{u}_{\mathrm{Unf}} q$ for some word $u \in (\Sigma \setminus \Sigma_k)^\star$.

THEOREM 3.1. *The asynchronous automaton $\widehat{\mathcal{A}_{\mathrm{Unf}}}$ satisfies $L(\widehat{\mathcal{A}_{\mathrm{Unf}}}) = L(\mathcal{A})$. Moreover the number of local states $Q_k$ in each process is polynomial in $|Q|$ where $|Q|$ is the number of states in $\mathcal{A}$; more precisely $|Q_k| \leqslant O(|Q|^d)$ where $d = (2.|\Sigma| + 2)^{|\Sigma|+1}$.*

### 3.3 Sketch of proof

By induction on the structure of the unfolding it is not difficult to check the following first property (see Appendix A).

LEMMA 3.2. *The mapping $\beta_{\mathrm{Unf}}$ is a morphism from the unfolding $\mathcal{A}_{\mathrm{Unf}}$ to $\mathcal{A}$. Moreover for all $u \in L(\mathcal{A})$ there exists $v \in L(\mathcal{A}_{\mathrm{Unf}})$ such that $v \sim u$.*

The proof of Theorem 3.1 relies on an intermediate asynchronous automaton $\overline{\mathcal{A}_{\mathrm{Unf}}}$ over some extended independence alphabet $(\overline{\Sigma}, \overline{\|})$. We consider the alphabet $\overline{\Sigma} = \Sigma \cup \{(a, k) \in \Sigma \times K \mid a \notin \Sigma_k\}$ provided with the independence relation $\overline{\|}$ such that $a \,\overline{\|}\, b$ iff $a \| b$, $a \,\overline{\|}\, (b, k)$ iff $a \in \Sigma_k$, and $(a, k) \,\overline{\|}\, (b, k')$ iff $k = k'$ for all actions $a, b \in \Sigma$ and all processes $k, k' \in K$. For each process $k \in K$ we put $\overline{\Sigma_k} = \Sigma_k \cup \{(a, k) \mid a \in \Sigma \setminus \Sigma_k\}$. It is easy to check that $\left(\overline{\Sigma_k}\right)_{k\in K}$ is a distribution of $(\overline{\Sigma}, \overline{\|})$. Now $\overline{\mathcal{A}_{\mathrm{Unf}}}$ shares with $\widehat{\mathcal{A}_{\mathrm{Unf}}}$ its local states $Q_k$ and its initial state. A global state $(q_k)_{k\in K}$ is final if there exists a final state $q \in F_{\mathrm{Unf}}$ of the unfolding such that $q_k = q$ for all $k \in K$. For each action $a \in \Sigma$ its transition relation $\overline{\partial}_a$ is such that $((q_k)_{k\in \mathrm{Loc}(a)}, (q'_k)_{k\in \mathrm{Loc}(a)}) \in \overline{\partial}_a$ if there exists a transition $q \xrightarrow{a}_{\mathrm{Unf}} q'$ such that $q_k = q$ and $q'_k = q'$ for all $k \in \mathrm{Loc}(a)$. Moreover for each internal action $(a, k) \in \overline{\Sigma} \setminus \Sigma$, we put $(q, q') \in \overline{\partial}_{(a,k)}$ if $q \xrightarrow{a}_{\mathrm{Unf}} q'$.

We consider the projection morphism $\rho : \overline{\Sigma}^\star \to \Sigma^\star$ such that $\rho(\varepsilon) = \varepsilon$, $\rho(u.a) = \rho(u).a$ if $a \in \Sigma$, and $\rho(u.a) = \rho(u)$ if $a \in \overline{\Sigma} \setminus \Sigma$. It is not difficult to prove that $L(\mathcal{A}_{\mathrm{Unf}}) \subseteq \rho(L(\overline{\mathcal{A}_{\mathrm{Unf}}}))$ and $\rho(L(\overline{\mathcal{A}_{\mathrm{Unf}}})) = L(\widehat{\mathcal{A}_{\mathrm{Unf}}})$. These two basic properties do not rely on the particular structure of $\mathcal{A}_{\mathrm{Unf}}$: They hold actually for any automaton.

On the contrary the proof of the next lemma is very technical and tedious and relies on the particular construction of boxes and triangles (see Appendix B for some details).

LEMMA 3.3. *For each box $\square_{T,q} = (\mathcal{B}_{T,q}, \beta_{T,q})$ we have $\rho(L(\overline{\mathcal{B}_{T,q}})) \subseteq [L(\mathcal{B}_{T,q})]$.*

We can now conclude. By Lemma 3.2 we have $[L(\mathcal{A}_{\mathrm{Unf}})] = L(\mathcal{A})$. On the other hand the two basic properties show that $[L(\mathcal{A}_{\mathrm{Unf}})] \subseteq L(\widehat{\mathcal{A}_{\mathrm{Unf}}})$. Conversely Lemma 3.3 yields $L(\widehat{\mathcal{A}_{\mathrm{Unf}}}) = \rho(L(\overline{\mathcal{A}_{\mathrm{Unf}}})) \subseteq [L(\mathcal{A}_{\mathrm{Unf}})]$. Therefore $L(\mathcal{A}) = L(\widehat{\mathcal{A}_{\mathrm{Unf}}})$.

## Conclusion and future work

We have presented a polynomial algorithm for the construction of non-deterministic asynchronous automata from regular trace languages. We have shown that this new unfolding method improves the complexity of known techniques in terms of the number of local and global states. Several variations of our approach lead to analogous complexity results. We have selected here the simplest version to analyse. But it might not be the more efficient in practice.

Interestingly this unfolding method can be adapted to the implementation of any globally-cooperative compositional high-level message sequence charts as investigated in [5]. At present we are developping more involved unfolding techniques in order to construct deterministic safe asynchronous automata [13]. We are also investigating a possible extension of our unfolding technique to infinite traces [4].

## References

1. Bednarczyk M.A.: *Categories of Asynchronous Systems*. PhD thesis in Computer Science (University of Sussex, 1988)
2. Cori R., Métivier Y. and Zielonka W.: *Asynchronous mappings and asynchronous cellular automata*. Inform. and Comput. **106** (1993) 159–202
3. Diekert V. and Rozenberg G.: *The Book of Traces*. (World Scientific, 1995)
4. Gastin P. and Petit A.: *Asynchronous Automata for Infinite Traces*. ICALP, LNCS **623** (1992) 583–594
5. Genest B., Muscholl A. and Kuske D.: *A Kleene Theorem for a Class of Communicating Automata with Effective Algorithms*. DLT, LNCS **3340** (2004) 30–48
6. Klarlund N., Mukund M. and Sohoni M.: *Determinizing Asynchronous Automata*. ICALP, LNCS **820** (1994) 130–141
7. Morin R.: *Concurrent Automata vs. Asynchronous Systems*. MFCS, LNCS (2005) – To appear
8. Mukund M., Narayan Kumar K. and Sohoni M.: *Synthesizing distributed finite-state systems from MSCs*. CONCUR, LNCS **1877** (2000) 521–535
9. Mukund M. and Sohoni M.: *Gossiping, Asynchronous Automata and Zielonka's Theorem*. Report TCS-94-2, SPIC Science Foundation (Madras, India, 1994)
10. Muscholl A.: *On the complementation of Büchi asynchronous cellular automata*. ICALP, LNCS **820** (1994) 142–153
11. Ochmański E.: *Regular behaviour of concurrent systems*. Bulletin of the EATCS **27** (Oct. 1985) 56–67
12. Pighizzini G.: *Synthesis of Nondeterministic Asynchronous Automata*. Algebra, Logic and Applications, vol. **5** (1993) 109–126
13. Ştefănescu A., Esparza J. and Muscholl A.: *Synthesis of distributed algorithms using asynchronous automata*. CONCUR, LNCS **2761** (2003) 20–34
14. Thiagarajan P.S.: *Regular Event Structures and Finite Petri Nets: A Conjecture*. Formal and Natural Computing, LNCS **2300** (2002) 244–256
15. Zielonka W.: *Notes on finite asynchronous automata*. RAIRO, Theoretical Informatics and Applications **21** (Gauthiers-Villars, 1987) 99–135

# A Proof of Lemma 3.2

An immediate induction shows that for each box $\square_{T,q} = (\mathcal{B}_{T,q}, \beta_{T,q})$ and each triangle $\triangle_{T,q} = (\mathcal{T}_{T,q}, \tau_{T,q})$, the mappings $\beta_{T,q}$ and $\tau_{T,q}$ are morphisms from $\mathcal{B}_{T,q}$ to $\mathcal{A}_{T,q}$ and from $\mathcal{T}_{T,q}$ to $\mathcal{A}_{T,q}$ respectively. In particular $L(\mathcal{A}_{\mathrm{Unf}}) \subseteq L(\mathcal{A})$.

By induction on the size of $T$ we prove that for all paths $q \xrightarrow{u} q_1$ of $\mathcal{A}_{T,q}$ there exists an equivalent word $u' \sim u$ such that $\imath_{\square,T,q} \xrightarrow{u'} v$ is a path of $\mathcal{B}_{T,q}$ and $\beta_{T,q}(v) = q_1$. This property is trivial for the empty set $T = \emptyset$ because $\mathcal{A}_{\emptyset,q}$ and $\mathcal{B}_{\emptyset,q}$ are reduced to the state $q$. We shall distinguish two cases whether $T$ is connected or not.

Assume first that $T$ is a connected set of actions. We proceed by induction on the length of $u$. The property holds for the empty word because $\beta_{T,q}(\imath_{\square,T,q}) = q$. Let $q \xrightarrow{u} q_1 \xrightarrow{a} q_2$ be a path of $\mathcal{A}_{T,q}$. By induction there is $u' \sim u$ such that $\imath_{\square,T,q} \xrightarrow{u'} v$ is a path in $\mathcal{B}_{T,q}$ and $\beta_{T,q}(v) = q_1$. Then, by construction, the state $v = (w, T, q', k)$ comes from some triangle $\triangle_{T,q'}$. Furthermore $w$ comes from a box $\square_{T'',q''}$ inserted in $\triangle_{T,q'}$: We have $w = (w'', T'', q'', k'')$. We distinguish several cases.

1. If $a \in T \setminus T''$ and $|T \setminus T''| = 1$. Then $(w, a)$ belongs to $\textsc{Missing}(T, q', q_2)$. Consequently Line 25 of Alg. 5 shows that $v \xrightarrow{a} (\imath_{\triangle,T,q_2}, T, q_2, k')$ for some integer $k'$ and $\beta_{T,q}(\imath_{\triangle,T,q_2}, T, q_2, k') = \tau_{T,q_2}(\imath_{\triangle,T,q_2}) = q_2$.
2. If $a \in T \setminus T''$ and $|T \setminus T''| \neq 1$. Then Line 13 of Alg. 1 shows that $w \xrightarrow{a} w'$ with $w' = (\imath_{\square,T''\cup\{a\},q_2}, T'' \cup \{a\}, q_2, k')$ for some integer $k'$ is a transition of $\mathcal{T}_{T,q'}$ and $\tau_{T,q'}(w') = q_2$. Consequently $v \xrightarrow{a} (w', T, q', k)$ is a transition of $\mathcal{B}_{T,q}$ and $\beta_{T,q}(w', T, q', k) = q_2$ (see Line 14 of Alg. 5).
3. If $a \in T''$. By construction the path $\imath_{\square,T,q} \xrightarrow{u'} v$ of $\mathcal{B}_{T,q}$ consists of the sequence of transitions $\imath_{\square,T,q} \xrightarrow{u_1} v_1 \xrightarrow{u_2} v$ such that $u_1.u_2 = u'$, $v_1 = (w_1, T, q', k)$, $w_1 = (\imath_{\square,T'',q''}, T'', q'', k'')$ and all states $v_2$ reach along the path $v_1 \xrightarrow{u_2} v$ come from the same box $\square_{T'',q''}$ of the same triangle $\triangle_{T,q'}$ that is $v_2$ is some tuple $((w_2, T'', q'', k''), T, q', k)$. Consequently each action $b$ that occurs in $u_2$ belongs to $T''$: It follows that $q'' \xrightarrow{u_2} q_1 \xrightarrow{a} q_2$ is a path of $\mathcal{A}_{T'',q''}$. By induction there is an equivalent word $u_2' \sim u_2.a$ such that $\imath_{\square,T'',q''} \xrightarrow{u_2'} w'$ is a path of $\mathcal{B}_{T'',q''}$ and $\beta_{T'',q''}(w') = q_2$. Consequently $v_1 \xrightarrow{u_2'} ((w', T'', q'', k''), T, q', k)$ is a path of $\mathcal{B}_{T,q}$ and $\beta_{T,q}((w', T'', q'', k''), T, q', k) = q_2$ (see Line 12 of Alg. 1 and Line 14 of Alg. 5).

Suppose now that $T$ is an unconnected set of actions. Let $q \xrightarrow{u} q_1$ be a path of $\mathcal{A}_{T,q}$, $T_1$ be the connected component that contains the least action of $T$ and $T_2 = T \setminus T_1$. If $u|T_1 = \varepsilon$ then $q \xrightarrow{u} q_1$ is also a path of $\mathcal{A}_{T_2,q}$. Consequently, by induction there exists $u' \sim u$ such that $\imath_{\square,T_2,q} \xrightarrow{u'} w$ is a path of $\mathcal{B}_{T_2,q}$ and $\beta_{T_2,q}(w) = q_1$. It follows by Line 33 of Alg. 5 that $\imath_{\square,T,q} \xrightarrow{u'} (w, T_2, q, 1)$ is a path of $\mathcal{B}_{T,q}$ and $\beta_{T,q}(w, T_2, q, 1) = q_1$. If $u|T_1 = a.u_1$ and $u|T_2 = u_2$ then $q \xrightarrow{u_2} q_2 \xrightarrow{a} q_3 \xrightarrow{u_1} q_1$ is also a path of $\mathcal{A}_{T,q}$ because $u_2.a.u_1 \sim u$ and $\mathcal{A}$ satisfies ID. Moreover $q \xrightarrow{u_2} q_2$ is a path of $\mathcal{A}_{T_2,q}$ and $q_3 \xrightarrow{u_1} q_1$ is a path of $\mathcal{A}_{T_1,q_3}$. Consequently, by induction, there exists $u_2' \sim u_2$ such that $\imath_{\square,T_2,q} \xrightarrow{u_2'} w_2$ is a path of $\mathcal{B}_{T_2,q}$ and $\beta_{T_2,q}(w_2) = q_2$, and

on other hand, there exists $u'_1 \sim u_1$ such that $\imath_{\Box,T_1,q_3} \xrightarrow{u'_1} w_1$ is a path of $\mathcal{B}_{T_1,q_3}$ and $\beta_{T_1,q_3}(w_1) = q_1$. Then Alg. 5 ensures that $\imath_{\Box,T,q} \xrightarrow{u'_2} (w_2, T_2, q, 1)$ is a path of $\mathcal{B}_{T,q}$, $\beta_{T,q}(w_2, T_2, q, 1) = q_2$ (Line 33), $(\imath_{\Box,T_1,q_3}, T_1, q_3, k) \xrightarrow{u'_1} (w_1, T_1, q_3, k)$ is a path of $\mathcal{B}_{T,q}$ for some integer $k$ and $\beta_{T,q}(w_1, T_1, q_3, k) = q_1$ (Line 40). Finally we have $(w_2, T_2, q, 1) \xrightarrow{a} (\imath_{\Box,T_1,q_3}, T_1, q_3, k)$ (Line 35 and 41). It follows that $\imath_{\Box,T,q} \xrightarrow{u'_2.a.u'_1} (w_1, T_1, q_3, k)$ is a path of $\mathcal{B}_{T,q}$, $\beta_{T,q}(w_1, T_1, q_3, k) = q_1$ and $u'_2.a.u'_1 \sim u$.

## B Proof sketch of Lemma 3.3

The complete proof of Lemma 3.3 requires about 20 pages of tedious technical details. In this appendix we present the two main ideas that lead the argument. We need first to introduce some basic definitions and notations precisely.

*Some basic definitions and notations.* Let $\mathcal{A}$ be some automaton over $\Sigma$. A *path* of length $n \in \mathbb{N} \setminus \{0\}$ is a sequence of transitions $\left(q_i \xrightarrow{a_i} q'_i\right)_{i\in[1,n]}$ such that $q'_i = q_{i+1}$ for all integers $0 < i < n$. For all words $u \in \Sigma^\star$ we write $q \xrightarrow{u} q'$ to denote a path $\left(q_i \xrightarrow{a_i} q'_i\right)_{i\in[1,n]}$ where $q_1 = q$, $q'_n = q'$, and $u = a_1...a_n$. Then $q$ is called the domain of $q \xrightarrow{u} q'$ and $q'$ is called its codomain. A path of length 0 is simply a state $q$ of $\mathcal{A}$. Its domain and codomain are equal to $q$.

If $s$ and $s'$ are two paths such that the codomain of $s$ is the domain of $s'$ then the *product* $s \cdot s'$ is defined in a natural way: If the length of $s$ is 0 then $s \cdot s' = s'$; if the length of $s'$ is 0 then $s \cdot s' = s$; otherwise $s \cdot s'$ is the concatenation of $s$ and $s'$.

Note that if $s$ is a path of the length $l > 0$ then it is the product of two paths $s = s_1 \cdot s_2$ where the length of $s_1$ is 1 and the length of $s_2$ is $l - 1$. Moreover such a product is unique. This remark allows us to define mappings for paths inductively on the length.

*Projections of global states and executions.* Assume now that $\mathcal{A}$ is (the global system of) an asynchronous automaton over the distribution $(\Sigma_k)_{k\in K}$. Then a path of $\mathcal{A}$ is called an *execution*. For convenience we shall consider the component automata $(\mathcal{A}_k)_{k\in K}$ defined as follows: For each process $j \in K$, $\mathcal{A}_j = (Q_j, \imath_j, \Sigma_j, \longrightarrow_j, Q_j)$ where $q_j \xrightarrow{a}_j q'_j$ if there are $q = (q_k)_{k\in \mathrm{Loc}(a)}$ and $q = (q'_k)_{k\in \mathrm{Loc}(a)}$ such that $(q, q') \in \partial_a$. Note here that $j \in \mathrm{Loc}(a)$ since $a \in \Sigma_j$.

Now the *projection* $s|k$ of an execution $s$ of $\mathcal{A}$ onto a process $j \in K$ is a path of $\mathcal{A}_j$ defined inductively as follows:

- $s|j = q_j$ if $s$ is a path of length 0 that corresponds to the global state $(q_k)_{k\in K}$;
- $s|j = q_j \xrightarrow{a} q'_j \cdot (s'|j)$ if $s$ is the product $s = t \cdot s'$ where $t$ is a transition $(q_k)_{k\in K} \xrightarrow{a} (q'_k)_{k\in K}$ and $j \in \mathrm{Loc}(a)$.
- $s|j = s'|j$ if $s$ is the product $s = t \cdot s'$ where $t$ is a transition $q \xrightarrow{a} q'$ and $j \notin \mathrm{Loc}(a)$.

*Executions of extended asynchronous automata.* In the paper we define the *extended asynchronous automaton* $\overline{\mathcal{A}_{\mathrm{Unf}}}$ of the unfolding automaton $\mathcal{A}_{\mathrm{Unf}}$. This definition can naturally be generalized to any automaton. Let $\mathcal{A}$ be an automaton and $\overline{\mathcal{A}}$ the corresponding extended asynchronous automaton. We say that an execution $s = q \xrightarrow{u} q'$ of $\overline{\mathcal{A}}$ is *arched* if there are two states $v$ and $v'$ in $\mathcal{A}$ such that for all $k \in K$, $q|k = v$ and $q'|k = v'$. Noteworthy each execution that leads an extended asynchronous automaton $\overline{\mathcal{A}}$ from its global initial state to some global final state is arched.

We define now a function $\gamma$ that associates each action of $\overline{\Sigma}$ to the corresponding action of $\Sigma$ in a natural way: For all actions $(a, k) \in \overline{\Sigma} \setminus \Sigma$, $\gamma(a, k) = a$ and for all actions $a \in \Sigma$, $\gamma(a) = a$. As usual this map extends from actions to words and we get $\gamma : \overline{\Sigma}^\star \to \Sigma^\star$. We can also extend the mapping $\gamma$ as a function from paths of component automata $\mathcal{A}_k$ to paths of $\mathcal{A}$ as follows. For each sequence $s$ that is a path of some $\mathcal{A}_k$, we define $\gamma(s)$ inductively on the length of $s$ by

– $\gamma(s) = q$ if the length of $s$ is 0 and $s = q$.

– $\gamma(s) = q \xrightarrow{\gamma(a)} q' \cdot \gamma(s')$ if $s$ is a product $s = t \cdot s'$ where $t$ is a transition $q \xrightarrow{a} q'$.

Clearly if $s$ is an execution of $\overline{\mathcal{A}}$ and $k$ a process of $K$ then $s|k$ is a path of $\mathcal{A}_k$ and $\gamma(s|k)$ is a path of $\mathcal{A}$.

*Definitions associated to unfoldings.* Let $T$ be a non-empty subset of $\Sigma$. We consider the triangle $\mathcal{T}_{T,q} = (Q_{\triangle,T,q}, \imath_{\triangle,T,q}, \longrightarrow_{\triangle,T,q}, F_{\triangle,T,q})$. Let $v$ be a state from $\mathcal{T}_{T,q}$. By construction of $\mathcal{T}_{T,q}$, $v$ is a quadruple $(w, T', q', k')$ such that $w$ is a state from the box $\square_{T',q'}$ and $k' \in \mathbb{N}$. We say that the *box location* of $v$ is $l^\square(v) = (T', q', k')$. We define the *sequence of boxes* reached along a path $s = q \xrightarrow{u} q'$ in $\mathcal{T}_{T,q}$ as follows:

– If the length of $s$ is 0 and $s$ corresponds to state $q \in Q_{\triangle,T,q}$ then $\mathbb{L}^\square(s) = l^\square(q)$.

– If $s$ is a product $s = s' \cdot t$ where $t$ is the transition $q \xrightarrow{a} q'$ then two cases appear:
  - If $l^\square(q) = l^\square(q')$ then $\mathbb{L}^\square(s) = \mathbb{L}^\square(s')$;
  - If $l^\square(q) \neq l^\square(q')$ then $\mathbb{L}^\square(s) = \mathbb{L}^\square(s').l^\square(q')$

Similarly we define the sequence of triangles $\mathbb{L}^\triangle(s_1)$ reached by a path $s_1$ in a box $\mathcal{B}_{T_1,q_1}$ where $T_1$ is a non-empty *connected* set of actions and the sequence of boxes $\mathbb{L}^\square(s_2)$ reached by a path $s_2$ in a box $\mathcal{B}_{T_2,q_2}$ where $T_2$ is an *unconnected* set of actions.

*Two main properties of unfoldings.* The following proposition states that all processes behave similarly in an extended asynchronous automaton built from boxes or triangles.

PROPOSITION B.1. *Let $\mathcal{B}_{T_1,q_1}$ be a box with $T_1$ a non-empty* connected *set of actions, $\mathcal{B}_{T_2,q_2}$ be a box with $T_2$ an* unconnected *set of actions, and $\mathcal{T}_{T_3,q_3}$ be a triangle with $T_3$ a non-empty set of actions. Let $s_1$, $s_2$ and $s_3$ be* arched *executions of $\overline{\mathcal{B}}_{T_1,q_1}$, $\overline{\mathcal{B}}_{T_2,q_2}$ and $\overline{\mathcal{T}}_{T_3,q_3}$ respectively. Then:*

1. $\forall k, k' \in \mathrm{Loc}(T_1), \mathbb{L}^\triangle(\gamma(s_1|k)) = \mathbb{L}^\triangle(\gamma(s_1|k'))$;
2. $\forall k, k' \in K, \mathbb{L}^\square(\gamma(s_2|k)) = \mathbb{L}^\square(\gamma(s_2|k'))$;
3. $\forall k, k' \in K, \mathbb{L}^\square(\gamma(s_3|k)) = \mathbb{L}^\square(\gamma(s_3|k'))$.

**Proof.** Property 2 and Property 3 stem from the remark that $\mathcal{B}_{T_2,q_2}$ and $\mathcal{T}_{T_3,q_3}$ are made of boxes connected along a tree-like structure. The proof of Property 1 is more subtle. Let $a$ be an action of $T_1$ and $k, k'$ be two processes of $\mathrm{Loc}(a)$. We proceed by contradiction. Let $\mathcal{T}$ and $\mathcal{T}'$ be the first triangles that differ in $\mathbb{L}^\triangle(\gamma(s_1|k))$ and

$\mathbb{L}^{\triangle}(\gamma(s_1|k'))$. Let $c$ be the number of $a$-transitions that occur in $s_1$ just before $\gamma(s_1|k)$ and $\gamma(s_1|k')$ reach $\mathcal{T}$ and $\mathcal{T}'$. Since $s_1$ is arched, $\gamma(s_1|k)$ and $\gamma(s_1|k')$ have to meet eventually for the last state. Therefore $\gamma(s_1|k)$ and $\gamma(s_1|k')$ have to leave triangles $\mathcal{T}$ and $\mathcal{T}'$ respectively. Consequently, there is a $(c+1)^{th}$ $a$-transition $q \xrightarrow{a} q'$ in $s_1$. Moreover, this transition is such that $q|k$ is a state from $\mathcal{T}$ whereas $q|k'$ is a state from $\mathcal{T}'$, that is: $q|k \neq q|k'$. This contradicts the definition of $\overline{\partial}_a$. ■

PROPOSITION B.2. *Let $\mathcal{B}_{T,q}$ be a box. Let $s = q \xrightarrow{u} q'$ be an arched execution of $\overline{\mathcal{B}}_{T,q}$ with $q|k = w$ and $q'|k = w'$ for all $k \in K$. Then there is a word $v \in \Sigma^\star$ such that $v \sim \rho(u)$ and $w \xrightarrow{v} w'$ is a path of $\mathcal{B}_{T,q}$.*

**Proof.** We proceed by induction on the size of $T$. The case where $T = \emptyset$ is trivial because $\mathcal{B}_{\emptyset,q}$ consists of a single state $q$. Suppose that the property holds for all subsets $T' \subset T$ and all states $q' \in Q$. Assume first that $T$ is a connected set of actions. By Proposition B.1, we know that $\mathbb{L}^{\triangle}(\gamma(s|k)) = \mathbb{L}^{\triangle}(\gamma(s|k'))$ for all processes $k, k' \in \mathrm{Loc}(T_1)$. We claim first that we can find an other execution $s' = q \xrightarrow{u'} q'$ such that for *all* processes $k, k' \in K$, $\mathbb{L}^{\triangle}(\gamma(s'|k)) = \mathbb{L}^{\triangle}(\gamma(s'|k'))$ and moreover $\rho(u) = \rho(u')$. Let $\mathbb{L}^{\triangle}(\gamma(s'|k)) = \mathcal{T}_1 ... \mathcal{T}_n$ be the sequence of triangles visited by $\gamma(s'|k)$. We can split the execution $s'$ into several smaller arched executions $s_1, \ldots, s_n$ such that each execution $s_i$ is located within triangle $\mathcal{T}_i$. Similarly each execution $s_i$ can be split into several smaller arched executions $s'_1, ..., s'_m$ such that each execution $s'_j$ is located within a box $\mathcal{B}_j$ inserted in $\mathcal{T}_i$. Then we can conclude by applying the inductive hypothesis on each smaller box.

Assume finally that $T$ is an unconnected set of actions. By Proposition B.1, we know that for all processes $k, k' \in K$, $\mathbb{L}^{\square}(\gamma(s|k)) = \mathbb{L}^{\square}(\gamma(s|k'))$. Then we can conclude by applying the inductive hypothesis on the smaller boxes visited by $s$. ■

Lemma 3.3 follows now immediately: We have $\rho(L(\overline{\mathcal{B}}_{T,q})) \subseteq [L(\mathcal{B}_{T,q})]$.